REMARKS

Claims 1-22 are pending in the present application. Claims 1 and 8 were amended. No claims were added or cancelled. Reconsideration of the claims is respectfully requested.

I. 35 U.S.C. § 101

The Examiner has rejected claims 1-7 and 8-10 under 35 U.S.C. § 101 as being directed towards non-statutory subject matter. This rejection is respectfully traversed.

Claims 1 and 8 and have been amended to include the phrase "in a data processing system." Applicants respectfully submit that claims 1-7 and 8-10 are now in condition for allowance.

Therefore the rejection of claims 1-7 and 8-10 under 35 U.S.C. § 101 has been overcome.

II. 35 U.S.C. § 102, Anticipation, Claims 1-3, 6-13 and 16-22

The Examiner has rejected claims 1-13 and 16-22 under 35 U.S.C. § 102 as being anticipated by US Patent No. 6,438,560 to Loen (hereinafter called Loen). This rejection is respectfully traversed.

As to claims 1-13 and 16-22, the Office Action states:

Per claims 1 and 2:
Loen disclose:
- loading a class (fig. 2, element 42 and related discussion);
- inserting an immutability flag into the class (col. 6, line 16 "a particular field or flag defined for the object");
- determining whether the class is immutable (col. 2, lines 61-63 "determination is made as to whether a matching immutable object...has redundant content...requested immutable object"); and
- setting the immutability flag if the class is immutable (col. 6, lines 14-16 "Routine 50...determining first whether the object is immutable...based upon a particular field or flag").

Office Action dated November 3, 2004, page 3.

A prior art reference anticipates the claimed invention under 35 U.S.C. § 102 only if every element of a claimed invention is identically shown in that single reference,

arranged as they are in the claims. *In re Bond*, 910 F.2d 831, 832, 15 U.S.P.Q.2d 1566, 1567 (Fed. Cir. 1990). All limitations of the claimed invention must be considered when determining patentability. *In re Lowry*, 32 F.3d 1579, 1582, 32 U.S.P.Q.2d 1031, 1034 (Fed. Cir. 1994). Anticipation focuses on whether a claim reads on the product or process a prior art reference discloses, not on what the reference broadly teaches. *Kalman v. Kimberly-Clark Corp.*, 713 F.2d 760, 218 U.S.P.Q. 781 (Fed. Cir. 1983). The Loen reference cited by the Examiner does not anticipate the present invention as recited in claim 1, because Loen fails to teach each and every element of claim 1.

Amended independent claim 1, which is representative of independent claims 11 and 21 with regard to similarly recited subject matter, recites:

1.   A method in a data processing system for loading a class, the method comprising:
loading a class;
inserting an immutability flag into the class;
determining whether the class is immutable; and
setting the immutability flag if the class is immutable.

Independent claim 1 recites the feature of "inserting an immutability flag into the class." Loen does not teach this feature. The Examiner points to column 6, line 16 as teaching this feature. However to column 6, line 16 does not teach this feature. Column 6, line 14 through line 25 states:

Routine 50 begins in block 52 by determining first whether the object is immutable. Such a determination may be based upon a particular field or flag defined for the object, or by analyzing the context in which an object is created. In other instances, however, an object of a particular class type may be inherently immutable, with the create object routine utilized to create such an object being specific to that particular class type. As such, in some implementations block 52 may be omitted, it being understood that the object requested to be created must be previously known to be of an immutable type.

In the above cited passage, Loen teaches that when a request to create an object is received, it must determine if the object is immutable. This determination may be based on whether or not the object contains a certain flag or a certain field. However, no where

does this passage or any passage in Loen teach "inserting an immutability flag into the class." "[I]nserting an immutability flag into the class" is very different from checking to see if what flags and fields an object has. Therefore, Loen does not teach the feature of "inserting an immutability flag into the class," as recited in claim 1 of the present invention. Thus, Loen does not anticipate the present invention in claim 1, because Loen fails to teach each and every element of claim 1.

Furthermore, claim 1 also recites the feature of "setting the immutability flag if the class is immutable." Loen does not teach or suggest this feature. The Examiner points to column 6, line 14 through line 16, cited above, as teaching this feature. However, the above cited passage does not teach this feature. As was discussed above, the above cited passage of Leon teaches that when a request to create an object is received, a determination is made as to whether or not an object is immutable. This determination may be based on whether or not the object contains a certain flag or a certain field. Checking to see if an object contains a certain flag or field is not the same as "setting the immutability flag if the class is immutable." Therefore, Loen does not teach the feature of "setting the immutability flag if the class is immutable," as recited in claim 1 of the present invention. Thus, Loen does not anticipate the present invention in claim 1, because Loen fails to teach each and every element of claim 1.

Therefore, for all the reasons set forth above, Applicants submit that independent claims 1, 11 and 21 are allowable as Loen fails to teach each and every element of the claims. Claims 2, 3, 6, 7, 12, 13, 16 and 17 are dependent claims depending on independent claims 1 and 11. As Applicants have already demonstrated claims 1 and 11 to be in condition for allowance, Applicants respectfully submit that claims 2, 3, 6, 7, 12, 13, 16 and 17 are also allowable, at least by virtue of their dependency on allowable claims.

Additionally, the dependent claims also recite features not taught or suggested by Loen. For example, claims 2 and 12 recite the feature of "parsing the bytecode of the class." Loen does not teach or suggest this feature. While the Examiner has rejected this claim, the Examiner has failed to point to any part of Loen as teaching this feature. In fact, the Examiner does not make any reference to this feature at all. However, Loen does not teach "parsing the bytecode of the class." Loen does not refer to either parsing or bytecode anywhere. Therefore, Loen does not teach the feature of "parsing the bytecode

of the class," as recited in claims 2 and 12 of the present invention. Thus, Loen does not anticipate the present invention in claims 2 and 12, because Loen fails to teach each and every element of claims 2 and 12.

Furthermore, amended independent claim 8, which is representative of independent claims 18 and 22 with regard to similarly recited subject matter, recites:

8.   A method in a data processing system for invoking an object, the method comprising:
    receiving a request to invoke an object;
    examining an argument in the request;
    if the argument is an object, determining whether the object is immutable; and
    if the object is immutable, passing a reference to the object rather than a clone of the object.

Independent claim 8 recites the feature of "receiving a request to invoke an object." Loen does not teach this feature. The Examiner points to column 2, line 55 and column 5, line 8 through line 11 as teaching this feature. However, neither of these passages teaches this feature. Column 2, line 53 through line 65 and column 5, line 3 through line 14 state:

The invention addresses these and other problems associated with the prior art by providing an apparatus, program product, and method of processing a request to create an immutable object in which an existing immutable object is reused in appropriate circumstances to represent redundant data without the necessity for creating an additional immutable object. Specifically, prior to creating a new object in response to a request to create an immutable object, a determination is made as to whether a matching immutable object already exists that has redundant content with respect to the requested immutable object. If so, creation of a new object is inhibited, and a reference to the matching immutable object is returned in response to the request.

Computer 30 operates under the control of an operating system 40. and executes or otherwise relies upon various computer software applications, components, programs, objects, modules, data structures, etc. (e.g., class files 42, virtual machine 44, and object heap 46, among others). Moreover, various applications, components, programs, objects, modules, etc. may also execute on one or more processors in another computer

coupled to computer 30 via a network 38, e.g., in a distributed or client-server computing environment, whereby the processing required to implement the functions of a computer program may be allocated to multiple computers over a network.

Neither of the above cited passages teaches the feature of "receiving a request to invoke an object." Column 2, line 53 through line 65 teaches that Loen is directed to avoiding creating duplicates of immutable objects. Specifically Loen teaches receiving a request to create a new immutable object and checking to see if a matching immutable object exists with identical content and then returning a pointer to that object instead of creating a duplicate immutable object. However, receiving a request to create a new object is not the same as "receiving a request to invoke an object." Invoking an object involves calling an already existing object. Therefore a request to invoke an already existing object is not the same as a request to create a new object. Thus receiving a request to create a new object is not the same as "receiving a request to invoke an object." Therefore, Loen does not teach the feature of "receiving a request to invoke an object," as recited in claim 8 of the present invention. Thus, Loen does not anticipate the present invention in claim 8, because Loen fails to teach each and every element of claim 8.

Column 5, line 3 through line 14 of Loen also does not teach the feature of "receiving a request to invoke an object." Instead, the above cited passage teaches that computers need operating systems and software to operate and that various software and applications can operate on multiple computers in a network environment. This has nothing to do with "receiving a request to invoke an object." Therefore, Loen does not teach the feature of "receiving a request to invoke an object," as recited in claim 8 of the present invention. Thus, Loen does not anticipate the present invention in claim 8, because Loen fails to teach each and every element of claim 8.

Therefore, for all the reasons set forth above, Applicants submit that independent claims 8, 18 and 22 are allowable as Loen fails to teach each and every element of the claims. Claims 9, 10, 19 and 20 are dependent claims depending on independent claims 8 and 18. As Applicants have already demonstrated claims 8 and 18 to be in condition for allowance, Applicants respectfully submit that claims 9, 10, 19 and 20 are also allowable, at least by virtue of their dependency on allowable claims.

Additionally, the dependent claims also recite features not taught or suggested by Loen. For example, claims 9 and 19 recite the feature of "determining whether an immutability flag for the object is set." Loen does not teach this feature. As was discussed above, Loen does not teach the use of an immutability flag. Therefore, it follows that Loen does not teach "determining whether an immutability flag for the object is set," as recited in claims 9 and 19 of the present invention. Thus, Loen does not anticipate the present invention in claims 9 and 19, because Loen fails to teach each and every element of claims 9 and 19.

Furthermore, claims 10 and 20 recite the feature of "if the object is not immutable, passing a clone of the object as the argument." Loen does not teach this feature. The Examiner points to column 5, line 64 through line 66 as teaching this feature. However, the referenced passage does not teach this feature. Column 5, line 64 through line 66 states:

> Moreover, it is assumed that a request to create an immutable object expects the return of a "reference", or pointer, to an immutable object containing the desired data.

The above cited passage does not teach "if the object is not immutable, passing a clone of the object as the argument." The passage above teaches returning a reference or pointer to an immutable object. However, the passage does not teach or suggest the feature of "if the object is not immutable, passing a clone of the object as the argument." Nowhere does Loen teach or suggest creating a clone of an object. Therefore, it follows that Loen does not teach "if the object is not immutable, passing a clone of the object as the argument," as recited in claims 10 and 20 of the present invention. Thus, Loen does not anticipate the present invention in claims 10 and 20, because Loen fails to teach each and every element of claims 10 and 20.

Therefore, the rejection of claims 1-3, 6-13 and 16-22 under 35 U.S.C. § 102 has been overcome.

III. 35 U.S.C. § 103, Obviousness, Claims 4, 5, 14 and 15

The Examiner has rejected claims 4, 5, 14 and 15 under 35 U.S.C. § 103 as being unpatentable over Loen in view of US Publication No. 2002/0016864 to Brett (hereinafter called Brett). This rejection is respectfully traversed.

As to claims 4, 5, 14 and 15, the Office Action states:

> Loen does not explicitly disclose whether the class can be modified comprises determining whether all properties of the object are marked private and whether the class can be modified comprises determining whether there are any non-private methods that update properties of the class.

> However, Brett discloses in an analogous computer system whether the class can be modified comprises determining whether all properties of the object are marked private (page 1, paragraph 4 "functions are considered "private" or for use by only the object itself") and whether the class can be modified comprises determining whether there are any non-private methods that update properties of the class (page 1, paragraph 4 "functions can be declared "public" or available for use externally of the object").

> Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the method of having private and non-private or public variable within the class as taught by Brett into the method of reuse of immutable object at the run time as taught by Loen. The modification would be obvious because of one of ordinary skill in the art would be motivated to use private and/or public variables within the class to keep the data private and/or public in conversion for classes as suggested by Brett (page 3, paragraph 18).

Office Action dated November 3, 2004, page 6-7.

Applicants respectfully submit that the rejections of claims 4, 5, 14 and 15 under 35 U.S.C. § 103(a) are improper under 35 U.S.C. § 103(c). 35 U.S.C. § 103(c), effective November 29, 1999, reads as follows:

> (c) Subject matter developed by another person, which qualifies as prior art only under one or more of subsections (e), (f), and (g) of section 102 of this title, shall not preclude patentability under this section where the subject matter and the claimed invention were, at the time the invention was made, owned by the same person or subject to an obligation of assignment to the same person.

Applicants respectfully submit that the present application and Loen were, at the time the invention of the present application was made, commonly owned by

International Business Machines Corporation. *See* 1241 O.G. 97 (December 26, 2000), which states that an Applicant or Applicant's attorney may, pursuant to 35 U.S.C. §103(c), overcome a rejection by making a conspicuous statement that the application under examination and cited reference were commonly owned at the time the invention of the application under examination was made.

The present invention was filed on August 1, 2002, which is dated after November 29, 1999. In addition, the present invention and the Loen reference were, at the time the invention was made, owned by the same entity or subject to an obligation of assignment to the same entity, i.e., International Business Machines Corporation. Therefore, under 35 U.S.C. § 103(c), Loen is disqualified as prior art under 35 U.S.C. §103 against the claims of the present application. *See* MPEP § 2146.

In view of the above, Applicants respectfully request withdrawal of the rejections to claims 4, 5, 14 and 15 under 35 U.S.C. § 103(a)

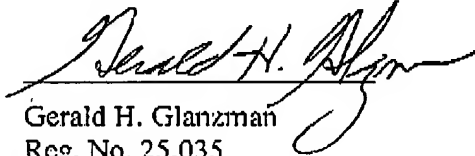Therefore, the rejection of claims 4, 5, 14 and 15 under 35 U.S.C. § 103 has been overcome.

IV.   Conclusion

It is respectfully urged that the subject application is patentable over the cited references and is now in condition for allowance.

The Examiner is invited to call the undersigned at the below-listed telephone number if in the opinion of the Examiner such a telephone conference would expedite or aid the prosecution and examination of this application.

DATE: *January 26, 2005*

Respectfully submitted,

Gerald H. Glanzman
Reg. No. 25,035
Yee & Associates, P.C.
P.O. Box 802333
Dallas, TX 75380
(972) 385-8777
Attorney for Applicants

GHG/bj

Page 14 of 14
Hogstrom et al.   10/041,127